Project Overview

Background

Water access is a critical issue in Tanzania, affecting millions of people. Despite significant investments in water infrastructure, many communities still face challenges in accessing clean and reliable water. The functionality of water pumps plays a crucial role in ensuring that communities have continuous access to this vital resource. However, a significant number of water pumps become non-functional over time due to various reasons such as poor maintenance, environmental conditions, and lack of resources.

Business Understanding:

Many water pumps in Tanzania are non-functional, leading to reduced access to clean water. Predicting pump functionality can help in timely maintenance and resource allocation and predicting pump functionality, the stakeholders can ensure better maintenance, leading to improved water access and health outcomes.

Loading and Merging the data

In [1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
```

In [2]:
```
df1 = pd.read_csv(r"C:\Users\ADMIN\Documents\phase3 data science project\0bf8bc
df1
```

Out[2]:

	id	status_group
0	69572	functional
1	8776	functional
2	34310	functional
3	67743	non functional
4	19728	functional
...	...	...
59395	60739	functional
59396	27263	functional
59397	37057	functional
59398	31282	functional
59399	26348	functional

59400 rows × 2 columns

In [3]:
```
df2 = pd.read_csv(r"C:\Users\ADMIN\Documents\phase3 data science project\702ddf
df2
```

Out[3]:

	id	amount_tsh	date_recorded	funder	gps_height	installer	longitude	lat
0	50785	0.0	2013-02-04	Dmdd	1996	DMDD	35.290799	-4.05
1	51630	0.0	2013-02-04	Government Of Tanzania	1569	DWE	36.656709	-3.30
2	17168	0.0	2013-02-01	NaN	1567	NaN	34.767863	-5.00
3	45559	0.0	2013-01-22	Finn Water	267	FINN WATER	38.058046	-9.41
4	49871	500.0	2013-03-27	Bruder	1260	BRUDER	35.006123	-10.95
...	...	...	...	...	...	...	...	
14845	39307	0.0	2011-02-24	Danida	34	Da	38.852669	-6.58
14846	18990	1000.0	2011-03-21	Hiap	0	HIAP	37.451633	-5.35
14847	28749	0.0	2013-03-04	NaN	1476	NaN	34.739804	-4.58
14848	33492	0.0	2013-02-18	Germany	998	DWE	35.432732	-10.58
14849	68707	0.0	2013-02-13	Government Of Tanzania	481	Government	34.765054	-11.22

14850 rows × 40 columns

In [4]:
```
df3 = pd.read_csv(r"C:\Users\ADMIN\Documents\phase3 data science project\49107
df3
```

Out[4]:

	id	amount_tsh	date_recorded	funder	gps_height	installer	longitude	latitude
0	69572	6000.0	2011-03-14	Roman	1390	Roman	34.938093	-9.856322
1	8776	0.0	2013-03-06	Grumeti	1399	GRUMETI	34.698766	-2.147466
2	34310	25.0	2013-02-25	Lottery Club	686	World vision	37.460664	-3.821329
3	67743	0.0	2013-01-28	Unicef	263	UNICEF	38.486161	-11.155298
4	19728	0.0	2011-07-13	Action In A	0	Artisan	31.130847	-1.825359
...	...	...	...	...	...	...	...	...
59395	60739	10.0	2013-05-03	Germany Republi	1210	CES	37.169807	-3.253847
59396	27263	4700.0	2011-05-07	Cefa-njombe	1212	Cefa	35.249991	-9.070629
59397	37057	0.0	2011-04-11	NaN	0	NaN	34.017087	-8.750434
59398	31282	0.0	2011-03-08	Malec	0	Musa	35.861315	-6.378573
59399	26348	0.0	2011-03-23	World Bank	191	World	38.104048	-6.747464

59400 rows × 40 columns

In [5]:
```
# Merge datasets on the 'id' column
df = pd.merge(pd.merge(df1, df2, on='id', how='outer'), df3, on='id', how='oute
df.head()
```

Out[5]:

	id	status_group	amount_tsh_x	date_recorded_x	funder_x	gps_height_x	installer_x	lon
0	69572	functional	NaN	NaN	NaN	NaN	NaN	
1	8776	functional	NaN	NaN	NaN	NaN	NaN	
2	34310	functional	NaN	NaN	NaN	NaN	NaN	
3	67743	non functional	NaN	NaN	NaN	NaN	NaN	
4	19728	functional	NaN	NaN	NaN	NaN	NaN	

5 rows × 80 columns

Data Understanding

In [6]:
```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 74250 entries, 0 to 74249
Data columns (total 80 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   id                       74250 non-null  int64  
 1   status_group             59400 non-null  object 
 2   amount_tsh_x             14850 non-null  float64
 3   date_recorded_x          14850 non-null  object 
 4   funder_x                 13981 non-null  object 
 5   gps_height_x             14850 non-null  float64
 6   installer_x              13973 non-null  object 
 7   longitude_x              14850 non-null  float64
 8   latitude_x               14850 non-null  float64
 9   wpt_name_x               14850 non-null  object 
 10  num_private_x            14850 non-null  float64
 11  basin_x                  14850 non-null  object 
 12  subvillage_x             14751 non-null  object 
 13  region_x                 14850 non-null  object 
 14  region_code_x            14850 non-null  float64
 15  district_code_x          14850 non-null  float64
 16  lga_x                    14850 non-null  object 
 17  ward_x                   14850 non-null  object 
 18  population_x             14850 non-null  float64
 19  public_meeting_x         14029 non-null  object 
 20  recorded_by_x            14850 non-null  object 
 21  scheme_management_x      13881 non-null  object 
 22  scheme_name_x            7758 non-null   object 
 23  permit_x                 14113 non-null  object 
 24  construction_year_x      14850 non-null  float64
 25  extraction_type_x        14850 non-null  object 
 26  extraction_type_group_x  14850 non-null  object 
 27  extraction_type_class_x  14850 non-null  object 
 28  management_x             14850 non-null  object 
 29  management_group_x       14850 non-null  object 
 30  payment_x                14850 non-null  object 
 31  payment_type_x           14850 non-null  object 
 32  water_quality_x          14850 non-null  object 
 33  quality_group_x          14850 non-null  object 
 34  quantity_x               14850 non-null  object 
 35  quantity_group_x         14850 non-null  object 
 36  source_x                 14850 non-null  object 
 37  source_type_x            14850 non-null  object 
 38  source_class_x           14850 non-null  object 
 39  waterpoint_type_x        14850 non-null  object 
 40  waterpoint_type_group_x  14850 non-null  object 
 41  amount_tsh_y             59400 non-null  float64
 42  date_recorded_y          59400 non-null  object 
 43  funder_y                 55765 non-null  object 
 44  gps_height_y             59400 non-null  float64
 45  installer_y              55745 non-null  object 
 46  longitude_y              59400 non-null  float64
 47  latitude_y               59400 non-null  float64
 48  wpt_name_y               59400 non-null  object 
 49  num_private_y            59400 non-null  float64
 50  basin_y                  59400 non-null  object 
 51  subvillage_y             59029 non-null  object 
```

```
 52  region_y                  59400 non-null  object
 53  region_code_y             59400 non-null  float64
 54  district_code_y           59400 non-null  float64
 55  lga_y                     59400 non-null  object
 56  ward_y                    59400 non-null  object
 57  population_y              59400 non-null  float64
 58  public_meeting_y          56066 non-null  object
 59  recorded_by_y             59400 non-null  object
 60  scheme_management_y       55523 non-null  object
 61  scheme_name_y             31234 non-null  object
 62  permit_y                  56344 non-null  object
 63  construction_year_y       59400 non-null  float64
 64  extraction_type_y         59400 non-null  object
 65  extraction_type_group_y   59400 non-null  object
 66  extraction_type_class_y   59400 non-null  object
 67  management_y              59400 non-null  object
 68  management_group_y        59400 non-null  object
 69  payment_y                 59400 non-null  object
 70  payment_type_y            59400 non-null  object
 71  water_quality_y           59400 non-null  object
 72  quality_group_y           59400 non-null  object
 73  quantity_y                59400 non-null  object
 74  quantity_group_y          59400 non-null  object
 75  source_y                  59400 non-null  object
 76  source_type_y             59400 non-null  object
 77  source_class_y            59400 non-null  object
 78  waterpoint_type_y         59400 non-null  object
 79  waterpoint_type_group_y   59400 non-null  object
dtypes: float64(18), int64(1), object(61)
memory usage: 45.9+ MB
None
```

In [7]:

```python
#Checking for missing values
print(df.isnull().sum())
```

```
id                             0
status_group               14850
amount_tsh_x               59400
date_recorded_x            59400
funder_x                   60269
                           ...
source_y                   14850
source_type_y              14850
source_class_y             14850
waterpoint_type_y          14850
waterpoint_type_group_y    14850
Length: 80, dtype: int64
```

In [8]:

```python
# Replace zero values in latitude, longitude, and gps_height with NaN
df['gps_height_x'].replace(0, pd.NA, inplace=True)
df['longitude_x'].replace(0, pd.NA, inplace=True)
df['latitude_x'].replace(0, pd.NA, inplace=True)
```

In [9]:
```
# Fill missing categorical values with the mode
categorical_columns = df.select_dtypes(include=['object']).columns
for column in categorical_columns:
    df[column].fillna(df[column].mode()[0], inplace=True)

# Fill missing numerical values with the median
numerical_columns = df.select_dtypes(include=['int64', 'float64']).columns
for column in numerical_columns:
    df[column].fillna(df[column].median(), inplace=True)
```

Feature Engineering:

In [10]:
```
# Convert date_recorded to datetime
df['date_recorded_y'] = pd.to_datetime(df['date_recorded_y'])

# Create a new column for the age of the pump
df['pump_age'] = df['date_recorded_y'].dt.year - df['construction_year_y']
df['pump_age'].replace({-pd.NA: pd.NA}, inplace=True)
```

In [11]:
```
# Drop columns that won't be used for modeling
df.drop(['id', 'wpt_name_x', 'num_private_x', 'recorded_by_x', 'scheme_name_x',
```

In [12]:
```
# The updated dataframe
print(df.head())
```

```
     status_group  amount_tsh_x              funder_x  gps_height_x  \
0      functional           0.0  Government Of Tanzania        1165.0
1      functional           0.0  Government Of Tanzania        1165.0
2      functional           0.0  Government Of Tanzania        1165.0
3  non functional           0.0  Government Of Tanzania        1165.0
4      functional           0.0  Government Of Tanzania        1165.0

  installer_x  longitude_x  latitude_x       basin_x subvillage_x   region_x  
\
0         DWE    35.011381    -5.04975  Lake Victoria      Shuleni  Shinyanga
1         DWE    35.011381    -5.04975  Lake Victoria      Shuleni  Shinyanga
2         DWE    35.011381    -5.04975  Lake Victoria      Shuleni  Shinyanga
3         DWE    35.011381    -5.04975  Lake Victoria      Shuleni  Shinyanga
4         DWE    35.011381    -5.04975  Lake Victoria      Shuleni  Shinyanga

   ... water_quality_y  quality_group_y    quantity_y quantity_group_y  \
0  ...            soft             good        enough           enough
1  ...            soft             good  insufficient     insufficient
2  ...            soft             good        enough           enough
3  ...            soft             good           dry              dry
4  ...            soft             good      seasonal         seasonal

              source_y         source_type_y source_class_y  \
0               spring                spring    groundwater
1  rainwater harvesting  rainwater harvesting        surface
2                  dam                   dam        surface
3          machine dbh              borehole    groundwater
4  rainwater harvesting  rainwater harvesting        surface

             waterpoint_type_y  waterpoint_type_group_y pump_age
0           communal standpipe       communal standpipe     12.0
1           communal standpipe       communal standpipe      3.0
2  communal standpipe multiple       communal standpipe      4.0
3  communal standpipe multiple       communal standpipe     27.0
4           communal standpipe       communal standpipe   2011.0

[5 rows x 75 columns]
```

Exploratory Data Analysis (EDA)

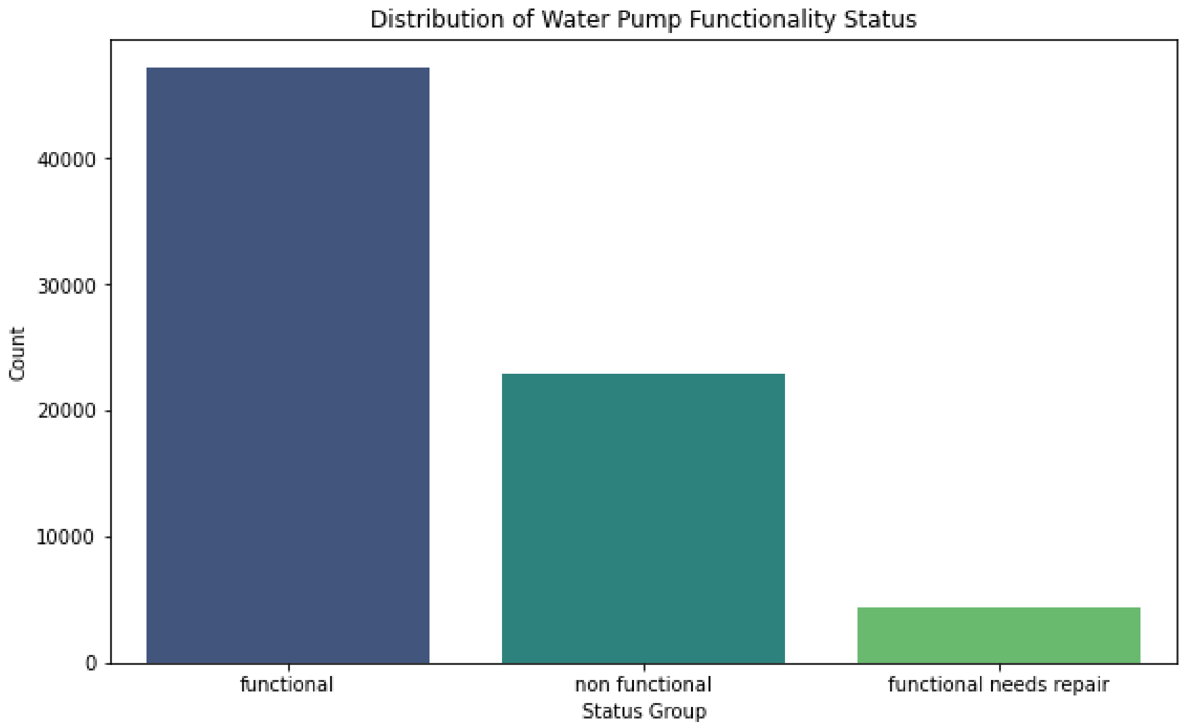
Visualizing the data to gain insights into the distribution and relationships.

In [13]:
```
print(df.info())
```

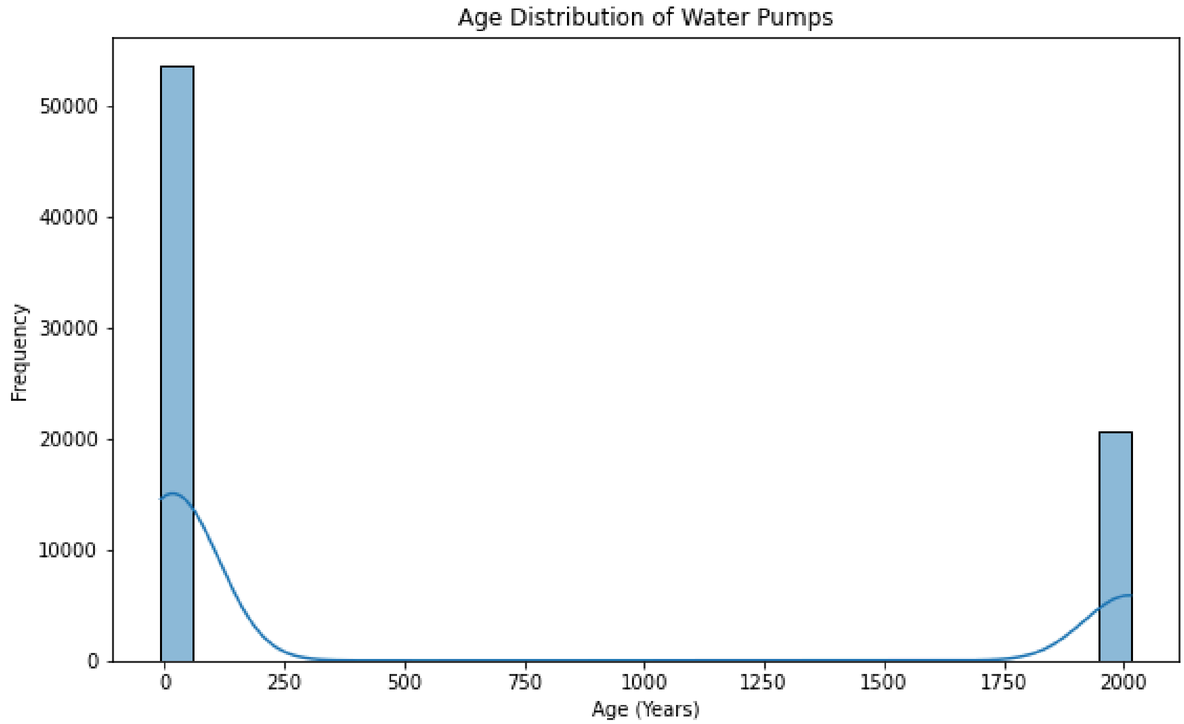
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 74250 entries, 0 to 74249
Data columns (total 75 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   status_group             74250 non-null  object
 1   amount_tsh_x             74250 non-null  float64
 2   funder_x                 74250 non-null  object
 3   gps_height_x             74250 non-null  float64
 4   installer_x              74250 non-null  object
 5   longitude_x              74250 non-null  float64
 6   latitude_x               74250 non-null  float64
 7   basin_x                  74250 non-null  object
 8   subvillage_x             74250 non-null  object
 9   region_x                 74250 non-null  object
 10  region_code_x            74250 non-null  float64
 11  district_code_x          74250 non-null  float64
 12  lga_x                    74250 non-null  object
 13  ward_x                   74250 non-null  object
 14  population_x             74250 non-null  float64
 15  public_meeting_x         74250 non-null  bool
 16  scheme_management_x      74250 non-null  object
 17  permit_x                 74250 non-null  bool
 18  construction_year_x      74250 non-null  float64
 19  extraction_type_x        74250 non-null  object
 20  extraction_type_group_x  74250 non-null  object
 21  extraction_type_class_x  74250 non-null  object
 22  management_x             74250 non-null  object
 23  management_group_x       74250 non-null  object
 24  payment_x                74250 non-null  object
 25  payment_type_x           74250 non-null  object
 26  water_quality_x          74250 non-null  object
 27  quality_group_x          74250 non-null  object
 28  quantity_x               74250 non-null  object
 29  quantity_group_x         74250 non-null  object
 30  source_x                 74250 non-null  object
 31  source_type_x            74250 non-null  object
 32  source_class_x           74250 non-null  object
 33  waterpoint_type_x        74250 non-null  object
 34  waterpoint_type_group_x  74250 non-null  object
 35  amount_tsh_y             74250 non-null  float64
 36  date_recorded_y          74250 non-null  datetime64[ns]
 37  funder_y                 74250 non-null  object
 38  gps_height_y             74250 non-null  float64
 39  installer_y              74250 non-null  object
 40  longitude_y              74250 non-null  float64
 41  latitude_y               74250 non-null  float64
 42  wpt_name_y               74250 non-null  object
 43  num_private_y            74250 non-null  float64
 44  basin_y                  74250 non-null  object
 45  subvillage_y             74250 non-null  object
 46  region_y                 74250 non-null  object
 47  region_code_y            74250 non-null  float64
 48  district_code_y          74250 non-null  float64
 49  lga_y                    74250 non-null  object
 50  ward_y                   74250 non-null  object
 51  population_y             74250 non-null  float64
```

```
 52  public_meeting_y        74250 non-null  bool
 53  recorded_by_y           74250 non-null  object
 54  scheme_management_y     74250 non-null  object
 55  scheme_name_y           74250 non-null  object
 56  permit_y                74250 non-null  bool
 57  construction_year_y     74250 non-null  float64
 58  extraction_type_y       74250 non-null  object
 59  extraction_type_group_y  74250 non-null  object
 60  extraction_type_class_y  74250 non-null  object
 61  management_y            74250 non-null  object
 62  management_group_y      74250 non-null  object
 63  payment_y               74250 non-null  object
 64  payment_type_y          74250 non-null  object
 65  water_quality_y         74250 non-null  object
 66  quality_group_y         74250 non-null  object
 67  quantity_y              74250 non-null  object
 68  quantity_group_y        74250 non-null  object
 69  source_y                74250 non-null  object
 70  source_type_y           74250 non-null  object
 71  source_class_y          74250 non-null  object
 72  waterpoint_type_y       74250 non-null  object
 73  waterpoint_type_group_y  74250 non-null  object
 74  pump_age                74250 non-null  float64
dtypes: bool(4), datetime64[ns](1), float64(18), object(52)
memory usage: 41.1+ MB
None
```

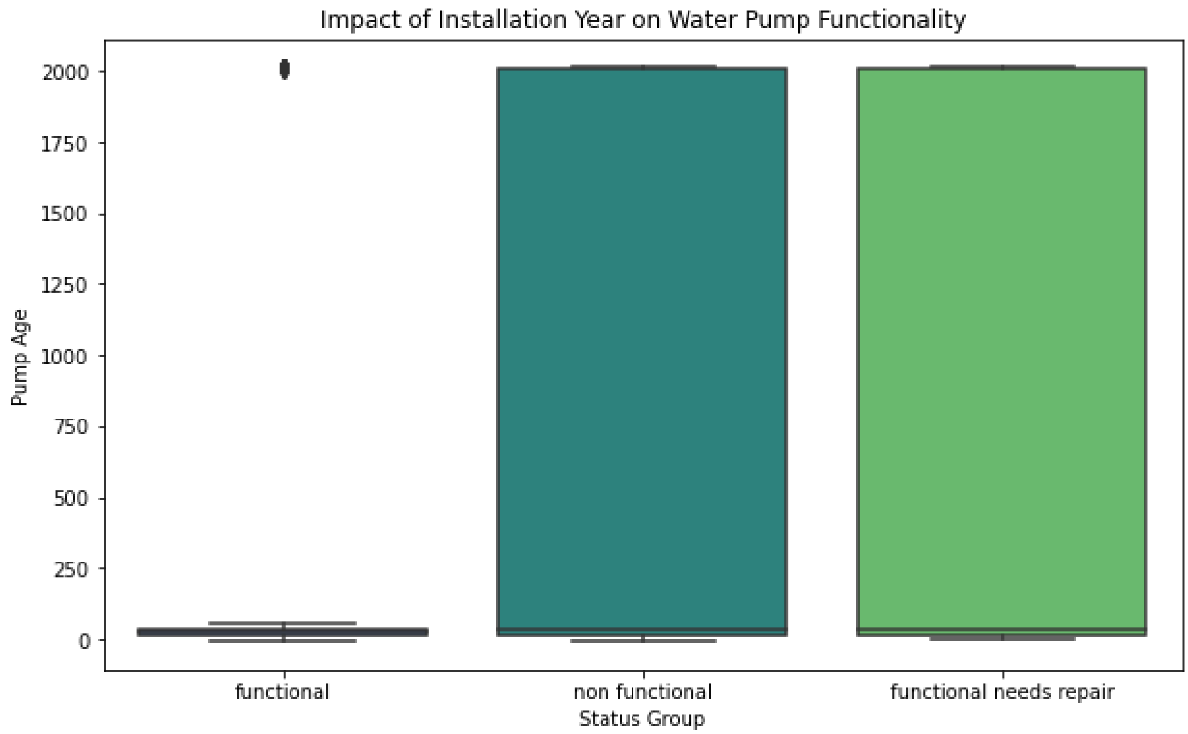
In [14]:
```
import matplotlib.pyplot as plt
import seaborn as sns

#Distribution of water pump functionality status
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='status_group', palette='viridis')
plt.title('Distribution of Water Pump Functionality Status')
plt.xlabel('Status Group')
plt.ylabel('Count')
plt.show()
```



In [15]:
```
#Age distribution of water pumps
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='pump_age', kde=True, bins=30)
plt.title('Age Distribution of Water Pumps')
plt.xlabel('Age (Years)')
plt.ylabel('Frequency')
plt.show()
```



In [16]:
```
#Impact of installation year on functionality
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='status_group', y='pump_age', palette='viridis')
plt.title('Impact of Installation Year on Water Pump Functionality')
plt.xlabel('Status Group')
plt.ylabel('Pump Age')
plt.show()
```



In [17]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer

# Sample the data to make it more manageable
#df = df.sample(frac=0.1, random_state=42)

# Split the data into features and target
X = df.drop(['status_group'], axis=1)
y = df['status_group']


categorical_features = X.select_dtypes(include=['object', 'bool']).columns
numerical_features = X.select_dtypes(include=['int64', 'float64']).columns

# Preprocessing pipelines
numerical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])

categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])

#Combine transformers into a single preprocessor
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_features),
        ('cat', categorical_transformer, categorical_features)
    ])
```

Model Building and Evaluation

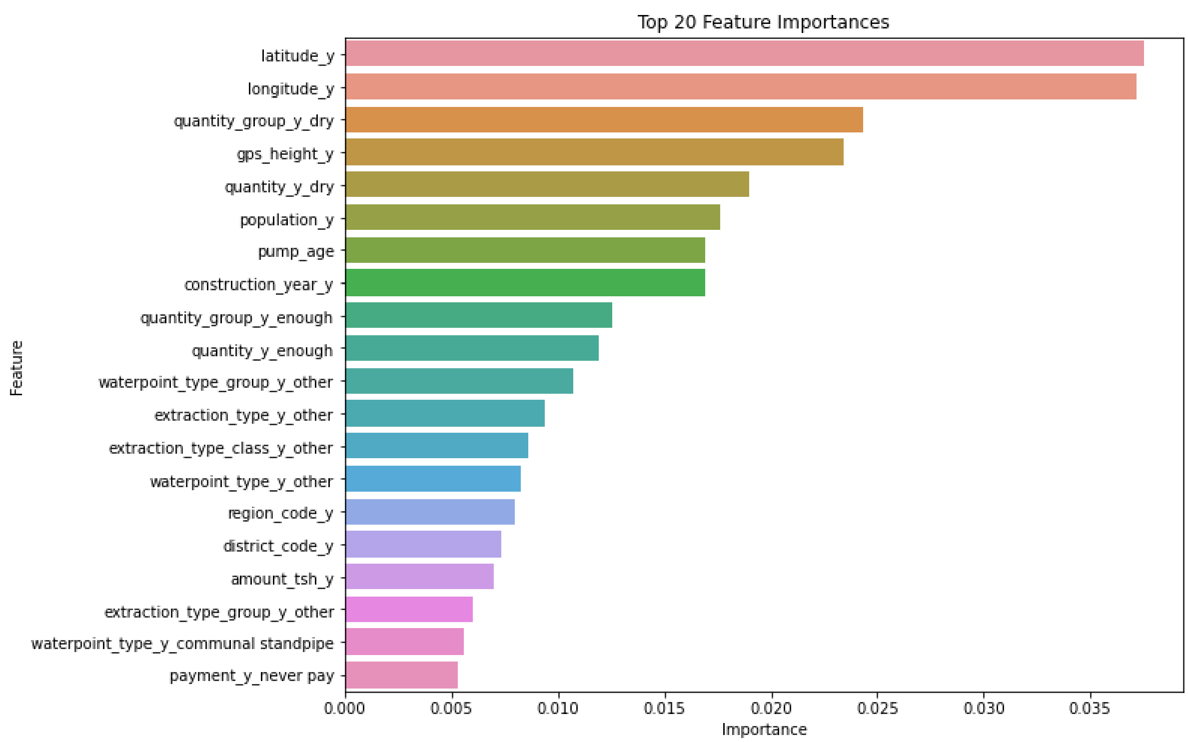
we will build and evaluate a RandomForestClassifier model.

In [18]:
```
# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, randor

# Define the model pipeline
model = Pipeline(steps=[('preprocessor', preprocessor),('classifier', RandomFor

# Train the model
model.fit(X_train, y_train)
# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
classification_report(y_test, y_pred)
```

Out[18]:
```
'                          precision    recall  f1-score   support\n\n
functional       0.87      0.93      0.90      9465\nfunctional needs repair
0.60      0.35      0.44       833\n         non functional       0.85
0.79      0.82      4552\n\n               accuracy
0.86     14850\n              macro avg       0.77      0.69      0.72     14
850\n           weighted avg       0.85      0.86      0.85     14850\n'
```

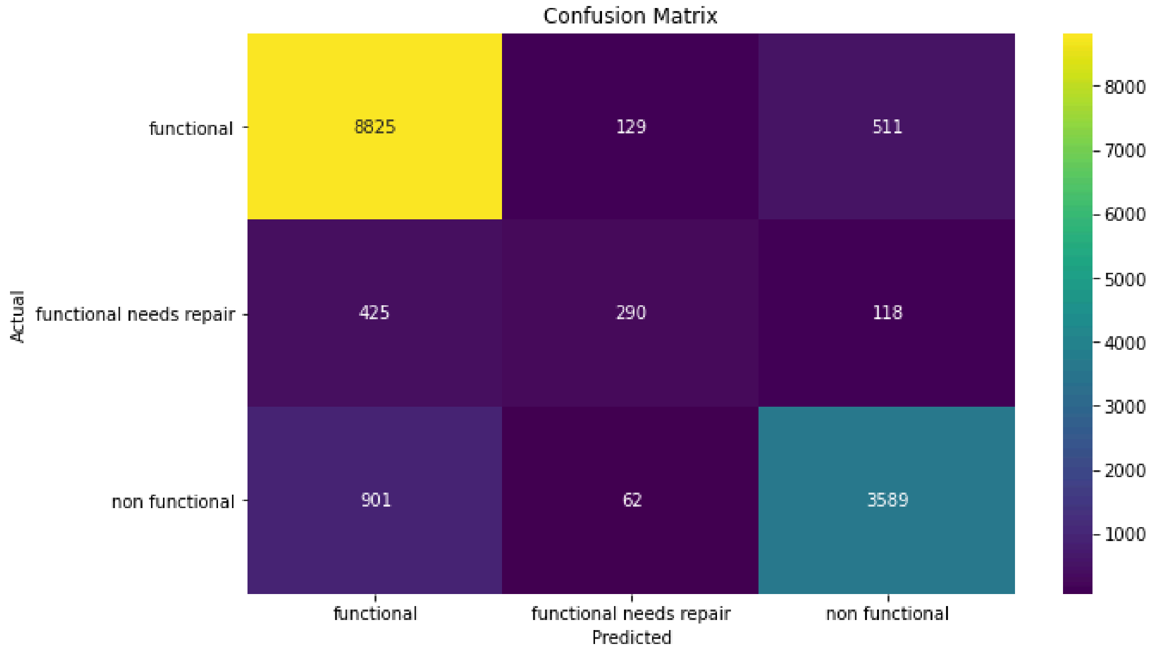
Visualization of Feature Importances

In [19]:
```
# Get feature importances from the random forest model
importances = model.named_steps['classifier'].feature_importances_
feature_names = preprocessor.transformers_[0][2].tolist() + \
                list(preprocessor.named_transformers_['cat'].named_steps['oneh

# Create a dataframe for the feature importances
feature_importances = pd.DataFrame({
    'feature': feature_names,
    'importance': importances
}).sort_values(by='importance', ascending=False)

# Plot the top 20 feature importances
plt.figure(figsize=(10, 8))
sns.barplot(data=feature_importances.head(20), x='importance', y='feature')
plt.title('Top 20 Feature Importances')
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.show()
```



In [20]:

```
from sklearn.metrics import confusion_matrix
import seaborn as sns

# confusion matrix
cm = confusion_matrix(y_test, y_pred, labels=model.classes_)

# Plot confusion matrix
plt.figure(figsize=(10, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='viridis', xticklabels=model.classes_
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```



Precision, Recall, and F1-Score

Functional Class:

> Precision (0.87): Out of all instances predicted as functional, 87% were correctly classified. Recall (0.93): Out of all actual functional pumps, 93% were correctly classified by the model. F1-Score (0.90): The harmonic mean of precision and recall for the functional class is 90%.

Functional Needs Repair Class:

> Precision (0.60): 60% of instances predicted as needing repair were correctly classified. Recall (0.35): Only 35% of actual pumps needing repair were correctly classified by the model. F1-Score (0.44): The F1-Score, which balances precision and recall, for this class is 44%.

Non-Functional Class:

> Precision (0.85): 85% of instances predicted as non-functional were correctly classified. Recall (0.79): 79% of actual non-functional pumps were correctly classified by the model. F1-Score (0.82): The F1-Score for the non-functional class is 82%.

Recommendations and Conclusion

To improve the performance and reliability of water pumps across Tanzania, consider the following:

1. Site Selection and Installation Geographical and Environmental Factors:

   > Elevation and Terrain: Install pumps in areas where the geographical features do not pose a risk of flooding or excessive wear due to rough terrain. Soil Quality: Ensure the soil around the installation site is stable and not prone to erosion or landslides. Water Table Levels: Select sites with a stable and sufficient water table to ensure continuous water availability.

2. Regular Maintenance and Monitoring

   > Scheduled Maintenance: Implement a routine maintenance schedule to check and service pumps regularly, preventing minor issues from becoming major problems. Remote Monitoring Systems: Utilize remote monitoring technology to track pump performance in real-time, enabling quick response to any emerging issues.

3. Data-Driven Decision Making

   > Data Collection: Continuously collect data on pump performance, environmental conditions, and usage patterns. Predictive Analytics: Use machine learning models to predict potential failures and plan proactive maintenance.

4. Quality of Materials and Installation Practices

   > High-Quality Materials: Use durable materials for pump components to reduce wear and tear and increase longevity. Skilled Labor: Ensure that the installation is carried out by skilled technicians to prevent early failures due to poor workmanship.

Factors to Consider for Best Results on Pump Performance and Reliability

1.Pump Type and Specifications: Choose pump types that are suited to the specific needs and conditions of the area (e.g., hand pumps for shallow wells, submersible pumps for deep wells).

2.Water Quality and Source Protection: Ensure that the water source is protected from contamination and that the pump is equipped to handle the specific water quality (e.g., pumps resistant to corrosion for saline water).

3.Infrastructure and Accessibility: Develop infrastructure around the pump site, such as access roads and storage facilities, to facilitate easy maintenance and repair work.

4.Environmental Impact Assessment: Conduct thorough environmental impact assessments before installation to ensure that the pump does not adversely affect local ecosystems.

In [ ]: